

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-324361

(43)Date of publication of application : 07.12.1993

(51)Int.Cl.

G06F 9/46

(21)Application number : 04-122985

(71)Applicant : FUJITSU LTD

(22)Date of filing : 15.05.1992

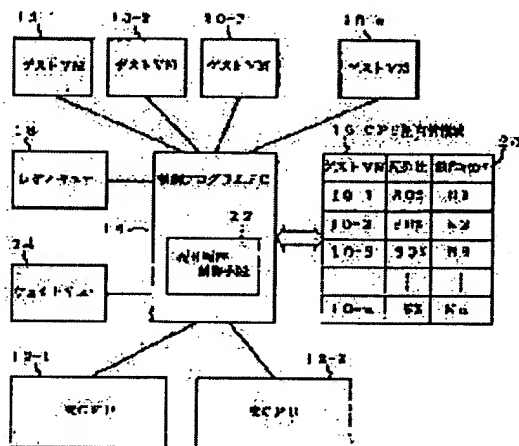
(72)Inventor : MATSUURA TAKEKAZU

## (54) CPU CONTROL SYSTEM IN VIRTUAL COMPUTER SYSTEM

## (57)Abstract:

PURPOSE: To maintain the distribution ratio of designated CPU resources and to efficiently utilize actual CPU even if the CPU distribution is unequal concerning the CPU control system of a virtual computer system where the CPU resources are distributed to plural virtual computers in the required ratio so as to permit the virtual computer to execute an operation.

CONSTITUTION: A system is provided in a CPU assignment information area 16, an operation counter 20 stores a time for digesting the distributed quantity of the designated CPU resources at every guest (VM10-1 to 10-n), the distribution order of the CPU resources in the guests VM10-1 to 10-n of a ready queue 18 is arranged in order of the operation times of the operation counter 20 from a small number by a distribution order control means 22, the CPU distribution of the guests VM10-1 to 10-n is executed in order of the distribution ratio from the large number when the operation times of the operation counter 20 are same and the CPU distribution of the guests VM10-1 to 10-n is executed in order of the operation times from the small number when the operation times are different.



## LEGAL STATUS

[Date of request for examination]

27.12.1995

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

2682770

[Date of registration]

08.08.1997

[Number of appeal against examiner's decision of rejection]

BEST AVAILABLE COPY

(19)日本国特許庁(J P)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平5-324361

(43)公開日 平成5年(1993)12月7日

(51)Int.Cl.<sup>5</sup>

G 0 6 F 9/46

識別記号

3 5 0

庁内整理番号

8120-5B

F I

技術表示箇所

審査請求 未請求 請求項の数6(全17頁)

(21)出願番号 特願平4-122985

(22)出願日 平成4年(1992)5月15日

(71)出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中1015番地

(72)発明者 松浦 豪一

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(74)代理人 弁理士 竹内 進 (外1名)

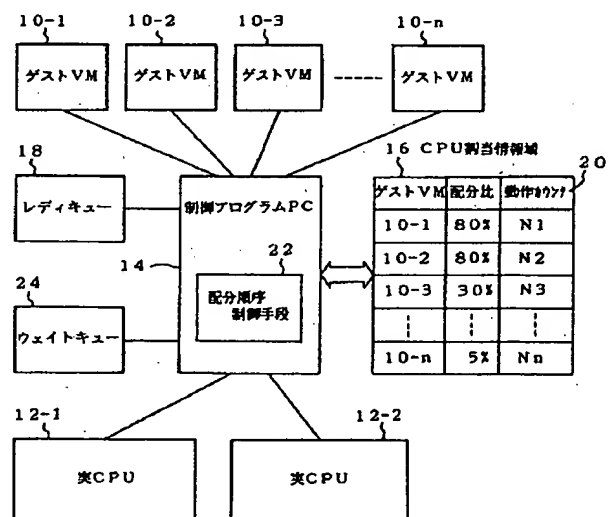
(54)【発明の名称】 仮想計算機システムのCPU制御方式

(57)【要約】

【目的】複数の仮想計算機に所望の割合でCPU資源を配分して仮想計算機を走行させる仮想計算機システムのCPU制御方式に関し、CPU配分が不均等であっても、指定したCPU資源の配分比を維持して実CPUの有効利用を図る。

【構成】CPU割当情報域16に設けられ、指定されたCPU資源の配分量を消化した回数をゲストVM10-1~10-n毎に動作カウンタ20で記憶し、レディキュー18上のゲストVM10-1~10-nのCPU資源の配分順序を配分順序制御手段22により動作カウンタ20の動作回数の小さい順番に並べ、動作カウンタ20の動作回数が同一の場合は配分比の大きい順番にゲストVM10-1~10-nのCPU配分を行い、動作回数が異なる場合には、動作回数の少ない順番にゲストVM10-1~10-nのCPU配分を行う。

本発明の原理説明図



## 【特許請求の範囲】

【請求項1】複数の実CPU(12-1, 12-2)を有する実計算機と、

該実計算機上で複数の仮想計算機(10-1~10-n)を動作可能とする制御プログラム(14)と、前記仮想計算機(10-1~10-n)毎に、割当てすべき前記実CPU(12-1, 12-2)の走行時間で決まるCPU資源の配分量を設定したCPU割当情報域(16)と、

前記CPU割当情報域(16)との対応関係を示すポインタ情報を、前記仮想計算機(10-1~10-n)のCPU資源の配分量が大きい程、高い優先順位をもつ配分順序となるように並べたレディキュー(18)と、を備え、前記制御プログラム(14)によりレディキュー(18)のポインタ情報で指定された順番に従って前記仮想計算機(10-1~10-n)に実CPU(12-1, 12-2)を割当て、前記CPU割当情報領域(16)の配分量に従って繰り返し動作させる仮想計算機システムのCPU制御方式に於いて、前記CPU割当情報域(16)に設けられ、指定されたCPU資源の配分量を消化した回数を仮想計算機(10-1~10-n)毎に記憶する動作カウンタ(20)と、

前記レディキュー(18)上の仮想計算機(10-1~10-n)のCPU資源の配分順序を前記動作カウンタ(20)の動作回数の小さい順番に並べる配分順序制御手段(22)と、

を備え、前記動作カウンタ(20)の動作回数が同一の場合は配分比の大きい順番に仮想計算機(10-1~10-n)のCPU配分を行い、動作回数が異なる場合には、動作回数の少ない順番に仮想計算機(10-1~10-n)のCPU配分を行うことを特徴とする仮想計算機システムのCPU制御方式。

【請求項2】請求項1記載の仮想計算機システムのCPU制御方式に於いて、CPU資源の配分時に待ち状態にある仮想計算機(10-i)のCPU割当情報域(16)を示すポインタ情報を取付けるウェイトキュー(24)を設け、前記制御プログラム(14)は前記仮想計算機(10-i)が動作可能状態となった時に前記ウェイトキュー(24)のポインタ情報を外して前記ウェイトキュー(18)に取付けることを特徴とする仮想計算機システムのCPU制御方式。

【請求項3】請求項2記載の仮想計算機システムのCPU制御方式に於いて、前記制御プログラム(14)は、仮想計算機(10-i)がCPU資源配分中に待ち状態になったことを検出した場合、該仮想計算機(10-i)が使い切っていない残り資源量をCPU割当用情報域(16)の対応位置に格納すると共に前記ウェイトキュー(24)に対応するCPU割当情報域(16)を示すポインタ情報を取付け、待ち状態の仮想計算機(10

-i)が動作可能になったことを検出した場合には、ポインタ情報を前記ウェイトキュー(24)から外して前記レディキュー(18)の前記動作カウンタ(20)の動作回数と配分比の大ききで決まる優先順位に従った位置に移してCPU資源の配分を再開させることを特徴とする仮想計算機システムのCPU制御方式。

【請求項4】請求項1記載の仮想計算機システムのCPU制御方式に於いて、前記配分順序制御手段(22)は、特定の仮想計算機(10-i)に対するCPU資源の配分が終了する毎に、前記動作カウンタ(20)の動作回数と各仮想計算機(10-1~10-n)の配分比を調べ、動作回数が同一の場合は配分比の大きい順番に並べ、動作回数が異なっている場合には、動作回数の少ないものを先頭側に移して配分比の大きい順番に並べ、次に動作回数の多いものを配分比の大きい順番に並べることを特徴とする仮想計算機システムのCPU制御方式。

【請求項5】請求項1記載の仮想計算機システムのCPU制御方式に於いて、前記動作カウンタ(20)は、CPU資源の配分量で規定された時間だけ仮想計算機(10-1~10-n)が走行する毎に1つカウントアップされ、該走行が完了した仮想計算機(10-i)を示すポインタ情報を前記配分順序制御手段(22)に従ったレディキュー(16)の位置に移すことを特徴とする仮想計算機システムのCPU制御方式。

【請求項6】請求項1記載の仮想計算機システムのCPU制御方式に於いて、前記レディキュー(18)には、現時点で優先順位が最も高い仮想計算機のCPU割当情報域(16)を示す先頭ポインタ情報と、現時点で優先順位が最も低い仮想計算機のCPU割当情報域(16)を示す末尾ポインタ情報とを格納し、先頭ポインタ情報で指定された仮想計算機のCPU資源の配分が終了する毎に、先頭ポインタ情報を末尾ポインタ情報の位置に移すことを特徴とする仮想計算機システムのCPU制御方式。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】本発明は、複数のCPUを有する実計算機上で複数の仮想計算機を動作可能とする制御プログラムによる仮想計算機システムのCPU制御方式に関し、特に、複数の仮想計算機に所望の割合でCPU資源を配分して仮想計算機を走行させる仮想計算機システムのCPU制御方式に関する。

【0002】仮想計算機モニタプログラム(VMモニタ)として知られた制御プログラムCPにより実計算機上で複数の仮想計算機(以下「ゲストVM」という)を動作可能とし、ゲストVMの各々がオペレーティングシステムOSで動作できる仮想計算機システムが大型コンピュータの利用技術の一つとして知られている。このような仮想計算機システムで、複数のゲストVMを複数の

3

CPU上で動作させる場合には、ゲストVMがCPU能力を有効に利用できるようにするため、各ゲストVMにCPU資源の配分量(配分比)を決め、この配分量に従って順番にゲストVMを走行させている。ゲストVMの走行順は、CPU資源の割当量の多い順番となる。

【0003】しかし、配分量が均等でなかった場合には、ゲストVMの走行を繰り返すうちに、割当量の少ないゲストVMの走行回数が増加するようになり、意図したCPU資源の配分量が達成できなくなり、この点の改善が望まれる。

【0004】

【従来の技術】図9は従来の仮想計算機システムを示したもので、制御プログラム(CP)14によって実CPU12を備えた実計算機上に複数のゲストVM10-1~10-nを動作可能としており、ゲストVM10-1~10-nはオペレーティングシステムOSを各々動作させることができる。

【0005】この場合、実計算機システムは単一のCPU12のみを備えており、予め定めた配分比に従ってゲストVM10-1~10-nに実CPU12を割当て順番に走行させている。制御プログラム14に対しては、CPU割当情報領域として処理制御ブロック(PCB; Process Control Block)16が設けられ、ゲストVM10-1~10-n毎に予め定めたCPU資源の配分比が格納されている。

【0006】ゲストVM10-1~10-nの走行順位は、レディキュー18により決められる。レディキュー18にはゲストVM10-1~10-nに対するCPU資源の配分順序が、配分比の大きい順番に従って設定されている。レディキュー18には、制御処理ブロック16のゲストVM10-1~10-nを指定するポインタ情報がCPU配分比で決まる優先順位に従って格納される。

【0007】実際には、現時点で最も優先順位の高い先頭ポインタ情報と最も優先順位の低い末尾ポインタ情報が格納されている。このため制御プログラム14は実CPU12が動作可能となった時に、レディキュー18の先頭ポインタで指定される処理制御ブロック16を参照して、CPUを割当てるゲストVMと配分比を知り、配分比で規定される時間に亘り実CPU12により指定されたゲストVMでOSを走行させる。

【0008】更にウェイトキュー24が設けられ、レディキュー18の優先順位に従ってCPU配分を受けたゲストVMが待ち状態にあるとき、このゲストVMをウェイトキュー24に取付け、次のゲストVMにCPU配分を行う。待ち状態にあったゲストVMが動作可能になると、ウェイトキュー24から外してレディキュー18の末尾ポインタ位置に取り付けてCPU配分を行う。

【0009】図10は図9の仮想計算機システムの配分処理を示したフローチャートであり、まずステップS1

4

でゲストVM10-1~10-nのレディキュー18へのポインタ情報の取り付けをCPU配分比で決まる順番に従って行ない、先頭ポインタ情報による処理制御ブロック16の内容から指定されるゲストVMを選択する。

【0010】続いてステップS2に進んで、レディキュー18に基づく優先順位に従って選択したゲストVMに実CPU12を配分し、配分比で規定される時間に亘り実CPU12に選択したゲストVMの動作を行わせる。ステップS2の配分済みと、ステップS3で配分が済んだゲストVMのポインタ情報を末尾ポインタ情報の位置に移し、次に優先順位の高かったポインタ情報が先頭ポインタ情報の位置に移し、再びステップS1に戻って次のゲストVMの動作を行う。

【0011】図11は4つのゲストVMを対象に単一の実CPUでCPU配分を行った場合の処理順を示す。

尚、4つのゲストVMの名称をVM1, VM2, VM3, VM4として示している。ここでゲストVM1~VM4のCPU資源の配分比は、処理制御ブロック16により

VM1	80%
VM2	10%
VM3	5%
VM4	5%

に予め設定されているものとする。

【0012】単一の実CPUで、複数のゲストVM1~4にCPU資源を配分する場合は、CPU資源の配分の順番をCPU配分比の大きい順番に図11のように設定し、指定された順序に従い指定された通りの配分比を得ることができる。

【0013】

【発明が解決しようとする課題】このような従来の仮想計算機システムにあつては、図9に示したように基本的には単一のCPU12の場合にのみ指定された配分比に従って厳密にCPU資源を分配することが可能である。しかしながら、図12のような複数の実CPU12-1, 12-2の場合には、指定された配分比に従って厳密にCPU資源を分配することができないという問題があった。

【0014】これは処理制御ブロック16を繋げるレディキュー18における優先順位が配分比の大きさによってのみ並べられていたためである。単一のCPUの場合には、配分比に従って並べることで、指定した配分比でCPU資源が分配できるが、複数のCPUの場合には、より小さい配分比のゲストVMがより大きい配分比のゲストVMより多くの回数CPU配分を受けることがあり、指定した配分比が得られなくなる。これを詳細に説明すると次のようになる。

【0015】まず図12の複数の実CPU12-1, 12-2の場合に、図9の単一の実CPU12の場合と同様、配分比の大きさで決まる優先順位に従って複数のゲ

ストVM10-1~10-nにCPU資源の配分を行うと、CPU資源の配分比が全て同じで且つ実CPUに対する割当てが均等になっている場合はCPU配分どおりに配分できる。

【0016】図13は実CPU12-1にゲストVM1, VM3, VM5, VM7の4つを割当て、実CPU12-2にゲストVM2, VM4, VM6, VM8の4つを割当てる均等割当てを行い、且つ番号の若い順番に優先順位を設定した場合であり、ゲストVM1~8に対するCPU配分の順序性が保たれたまま処理を繰り返すことができる。

【0017】しかし、CPU配分比が不均等な場合には、処理を繰り返すにつれて最初に指定したCPU配分の順序性が崩れ、指定したCPU資源の配分比が実現できなくなる。図14は5つのゲストVM1~5を対象に2台の実CPU12-1, 12-2のCPU資源の配分処理を示したもので、ゲストVM1~5の配分比は不均等であり、

VM1	80%
VM2	80%
VM3	30%
VM4	5%
VM5	5%

に定められているものとする。

【0018】最初のCPU配分の順序は配分比の大きい順に、

VM1 → VM2 → VM3 → VM4 → VM5

となっており、1回目は、この順番に従って実CPU12-1, 12-2のCPU資源の配分が行われる。即ち、実CPU12-1をゲストVM1に割当てると同時に、実CPU12-2をゲストVM2に割当てて処理を開始する。ゲストVM1とゲストVM2の配分比は共に80%と同じなので、この配分比80%で決まる割当時間終了の同時刻でゲストVM1とゲストVM2の配分を終る。

【0019】次に実CPU12-1をゲストVM3に割当て、同時に実CPU12-2をゲストVM4に割当てる。ゲストVM3の配分比は30%であり、ゲストVM4の配分比は5%であることから、先にゲストVM4の配分が終了し、実CPU12-2を最後のゲストVM5に割当てる。これが1回目のCPU配分である。ここでゲストVM4及びゲストVM5が配分を終了してもゲストVM3は以前として配分中にあり、このため先に配分が終了したゲストVM4とゲストVM5がレディキュー18に順序付けられてしまう。

【0020】2回目目のCPU配分は、1回目目のゲストVM5の配分が終了した時点で実CPU12-2をゲストVM1に割当てることで開始される。このとき実CPU12-1による1回目目のゲストVM3の配分はまだ終了していない。1回目目のゲストVM3の配分が終了する

と、実CPU12-1をゲストVM2に割当てて2回目目のCPU配分を行う。

【0021】このゲストVM3の1回目目の配分が終了した時点で確定した2回目目の配分順序は、  
VM1 → VM2 → VM4 → VM5 → VM3  
となっている。

【0022】同様にして2回目目の配分を終了した時点で確定した3回目目の配分順序は、  
VM1 → VM4 → VM5 → VM3 → VM2  
となる。

【0023】図15は、図14のCPU配分の様子を時間割合で示したもので、実CPU12-1, 12-2に対するゲストVM1~5の時間的な割当てが判る。このように実CPU12-1, 12-2に対して不均等な配分比をもつゲストVM1~5の配分処理を行うと、配分比の小さいゲストVM4やゲストVM5は配分比の大きいゲストVM3へのCPU配分が完了する前にCPU配分が完了してしまうため、次の配分処理では、ゲストVM3よりも早くCPUが配分され、このような処理を繰り返すと指定された配分どおりにCPUを配分できなくなる。

【0024】図16は図14のCPU配分を繰り返して飽和状態に至った時のCPU配分の指定量に対する実際の割当量を示したもので、指定量の少ないゲストVM4, VM5の割当量が増加し、指定量の大きいゲストVM1~VM3は割当量が少なめになる。一方、CPU配分が図13に示したように均等だったとしても、ゲストVMが走行中に待ち状態となった場合には、制御プログラム14が待ち状態にあるゲストVMの制御を取り上げてウェイトキュー24に取付け、次のゲストVMにCPU配分を行う。

【0025】ウェイトキュー24に取付けゲストVMが動作可能になると、制御プログラム14は、レディキュー18の末尾ポインタ位置に動作可能となったゲストVMを取付ける。このためゲストVMの待ち状態が生ずると、実CPUに対して不均等なCPU配分が一時的に発生し、指定された配分どおりにCPUを配分できなくなる。

【0026】本発明は、このような従来の問題点に鑑みてなされたもので、CPU配分が不均等であっても、指定したCPU資源の配分比を維持して実CPUの有効利用を図るようにした仮想計算機システムのCPU制御方式を提供することを目的とする。また本発明は、均等なCPU配分に従った処理途中でゲストVMに処理待ちが発生しても、指定したCPU資源の配分比を維持して実CPUの有効利用を図るようにした仮想計算機システムのCPU制御方式を提供することを目的とする。

【0027】

【課題を解決するための手段】図1は本発明の原理説明図である。まず本発明は、複数の実CPU12-1, 1

2-2を有する実計算機と、実計算機上で複数のゲストVM(仮想計算機)10-1~10-nを動作可能とする制御プログラム(CP)14と、ゲストVM10-1~10-n毎に、割当てるべき実CPU12-1,12-2の走行時間で決まるCPU資源の配分量を設定したCPU割当情報域(PCB)16と、CPU割当情報域16との対応関係を示すポインタ情報を、ゲストVM10-1~10-nのCPU資源の配分量が大きい程、高い優先順位をもつ配分順序となるように並べたレディキュー18とを備え、制御プログラム14によりレディキュー18のポインタ情報で指定された順番に従ってゲストVM10-1~10-nに実CPU12-1,12-2を割当て、CPU割当情報領域16の配分量に従って繰り返し動作させる仮想計算機システムのCPU制御方式を対象とする。

【0028】このような仮想計算機システムのCPU制御方式につき本発明にあっては、CPU割当情報域16に設けられ、指定されたCPU資源の配分量を消化した回数をゲストVM10-1~10-n毎に記憶する動作カウンタ20と、レディキュー18上のゲストVM10-1~10-nのCPU資源の配分順序を動作カウンタ20の動作回数の小さい順番に並べる配分順序制御手段22とを備え、動作カウンタ20の動作回数が同一の場合は配分比の大きい順番にゲストVM10-1~10-nのCPU配分を行い、動作回数が異なる場合には、動作回数の少ない順番にゲストVM10-1~10-nのCPU配分を行うことを特徴とする。

【0029】更に、CPU資源の配分時に待ち状態にある仮想計算機10-iのCPU割当情報領域16を示すポインタ情報を取付けるウェイトキュー24を設け、制御プログラム14はゲストVM10-iが動作可能状態となった時にウェイトキュー24のポインタ情報を外してウェイトキュー(18)に取付ける。また仮想計算機10-iがCPU資源配分中に待ち状態になったことを制御プログラム14が検出した場合には、待ち状態となったゲストVM10-iが使い切っていない残り資源量(残り時間)をCPU割当情報域16の対応位置に格納すると共にウェイトキュー24に対応するCPU割当情報域16を示すポインタ情報を取付け、待ち状態の仮想計算機10-iが動作可能になったことを検出した場合には、ポインタ情報をウェイトキュー24から外してレディキュー18の動作カウンタ20の動作回数と配分比の大きさに決まる優先順位に従った位置に移してCPU資源の配分を再開させる。

【0030】一方、制御プログラム14の配分順序制御手段22は、特定の仮想計算機10-iに対するCPU資源の配分が終了する毎に、動作カウンタ20の動作回数と各ゲストVM10-1~10-nの配分比を調べ、動作回数が同一の場合は配分比の大きい順番に並べ、動作回数が異なっている場合には、動作回数の少ないもの

を先頭側に移して配分比の大きい順番に並べ、次に動作回数の多いものを配分比の大きい順番に並べる。

【0031】また動作カウンタ20は、CPU資源の配分量で規定された時間だけゲストVM10-1~10-nが走行する毎に1つカウントアップされ、走行が終了したゲストVMを示すポインタ情報を配分順序制御手段22に従ったレディキュー16の位置に移す。さらにレディキュー18には、現時点で優先順位が最も高いゲストVMのCPU割当情報領域16を示す先頭ポインタ情報と、現時点で優先順位が最も低いゲストVMのCPU割当情報領域16を示す末尾ポインタ情報とを格納し、先頭ポインタ情報で指定されたゲストVMのCPU資源の割当量の配分が終了する毎に、先頭ポインタ情報を末尾ポインタ情報の位置に移す。

【0032】

【作用】このような構成を備えた本発明の仮想計算機システムのCPU制御方式によれば、複数のゲストVMについて予め定めたCPU資源の配分量(配分比)が不均等であっても、CPU配分の順序をゲストVMの動作回数と配分比の両方で決めているため、配分処理を繰り返しても最初に設定した順序性を維持することができ、複数の実CPUの割当てにより指定されたCPU配分通りに実際の配分ができ、実CPU資源を有効に利用できる。

【0033】例えば配分比の小さいゲストVMが配分を繰り返す内に優先順位が高くなり、配分比の大きなゲストVMの優先順位が低くなってきても、動作回数を求めて動作回数の少ないゲストVMを優先させるため、同じ動作回数となる全てのゲストVMの配分が終了しない限り、次の動作回数でのCPU配分には移行せず、配分比の少ないゲストVMの動作回数が配分比の大きなゲストVMの動作回数より多くなってしまうことを防止できる。

【0034】またCPU配分が均等な場合に、CPU配分中にゲストVMに待ち状態が起きて一時的にCPU配分が不均等になっても、動作可能状態となった場合には、動作回数と配分比に従った順番の配分位置に取付けられるため、次の動作回数に段階では不均等状態が解消され、指定した配分比を維持できる。

【0035】

【実施例】図2は本発明の仮想計算機のCPU制御方式が適用されるハードウェア構成を示した説明図である。図2において、12-1,12-2は実計算機を構成するCPU(「実CPU」ともいう)であり、主記憶制御装置(MCU)26を介して主記憶装置(MSU)28に接続している。主記憶装置28にはオペレーティングシステム(OS)30が設けられ、実計算機システムとしてはオペレーティングシステム30を実計算機12-1,12-2で実行する。

【0036】また、主記憶装置28には仮想計算機システムを実現するためVMモニタとして知られた制御プロ

グラム(CP)14が設けられている。制御プログラム14は実計算機システム上で仮想計算機としてのゲストVM10-1, 10-2, ... 10-nを動作可能とし、制御プログラム14で動作可能となったゲストVM10-1~10-nのそれぞれはオペレーティングシステム30を走行させることができる。

【0037】また、主記憶制御装置26にはチャネルプロセッサ(CHP)32を介してデバイスバス34が接続され、デバイスバス34には複数のデバイスインタフェース36-1, 36-2を設けている。この実施例では、デバイスインタフェース36-1に磁気ディスク装置38を接続している。図3は図2のハードウェア構成で実現される本発明の仮想計算機システムの一実施例を示した説明図である。

【0038】図3にあっては、制御プログラム14により5つのゲストVM10-1~10-5を実現して動作可能とする仮想計算機システムを例にとっている。制御プログラム14に対してはCPU割当て情報域として処理制御ブロック(PCB)16が設けられる。処理制御ブロック16はゲストVMの名称に対応して予め定めたCPU配分の配分比を設定しており、更に本発明にあっては、新たに動作カウンタ20を設けている。

【0039】ここで、ゲストVM10-1~10-5の名称は、説明を簡単にするためVM1~VM5として示す。また、CPU資源の配分比はパーセント表示としている。CPU配分比に従ったCPU資源の配分は実際には実CPU12-1または12-2の割当て時間となる。動作カウンタ20はゲストVM10-1~10-5に設定されたCPU資源の配分比に従った実CPU12-1または12-2の割り当てを受けて配分処理を1回消費するごとに1つカウントアップした値を動作回数として記憶する。この実施例にあっては動作カウンタ20の動作回数をゲストVM10-1~10-5に対応して動作回数N1~N5で示している。

【0040】更に、制御プログラム14に対してはレディキュー18とウェイトキュー24が設けられる。レディキュー18には制御プログラム14に設けた本発明による配分順序制御手段22で定めたゲストVM10-1~5に対するCPU資源の配分順序に従った情報が取り付けられる。具体的にレディキュー18には処理制御ブロック16のゲストVM10-1~5を示すポイント情報が取り付けられる。

【0041】配分順序制御手段22によるゲストVM10-1~5の決定ルールは次のルール1及びルール2に従う。

[ルール1]; ゲストVM10-1~5の動作カウンタの動作回数N1~N5の小さいものを優先する。

【0042】[ルール2]; ゲストVM10-1~5の中で配分比の大きい順番に並べる。

ここで、配分順序を決めるルール1とルール2の関係は

ルール1の方が優先度が高く、ルール1を満足した状態でルール2が適用される。具体的には、動作カウンタの動作回数N1~N5が同じであれば配分比の大きい順にレディキュー18にポイント情報を取り付ける。

【0043】また、ゲストVM10-1~10-5の動作カウンタ20の動作回数N1~N5が異なっている、例えばいずれか1つの動作カウンタ20の動作回数が他の動作回数より少なかった場合には動作回数の少ないゲストVMをレディキュー18の先頭に取り付け、動作回数の少ないゲストVMの割当てを優先する。レディキュー18の具体的な構成としては、配分順序制御手段22により前記ルール1, 2により決定されたゲストVM1~5のCPU配分の順番について、現時点で先頭に位置するゲストVMの先頭ポイント情報と現時点で最後に位置するゲストVMの末尾ポイント情報の2つを格納する。

【0044】そして、先頭ポイントで指定されるゲストVMに対するCPU配分が終了すると、先頭ポイントを外して末尾ポイントに移し、次に高い順位にあるゲストVMのポイント情報を先頭ポイントに移し、先頭ポイントで指定されたゲストVMに対するCPU配分を行い、これをシーケンシャルに繰り返す。一方、ウェイトキュー24は制御プログラム14により新たなゲストVMにCPU配分を行う際に、このゲストVMが待ち状態にあるとき、この待ち状態にあるゲストVMのポイント情報をレディキュー18の先頭ポイント位置から外してウェイトキュー24に取り付ける。

【0045】このウェイトキュー24へのポイント情報の取り付けによりレディキュー18の先頭ポイント位置には次に優先順位の高いゲストVMのポイント情報がくることから、次のゲストVMに対しCPU配分を行う。勿論、ウェイトキュー24には複数のゲストVMのポイントを取り付けることができ、最大5つのゲストVM10-1~5分の取付領域があればよい。

【0046】ウェイトキュー24に対し待ち状態にあるゲストVMのポイント情報を取り付けた後、制御プログラム14は待ち状態にあるゲストVMが動作可能になるか否か常時監視しており、動作可能になるとウェイトキュー24からポイント情報を取り外し、配分順序制御手段22のルール1及びルール2で決まる優先順位の位置となるようにレディキュー18にウェイトキュー24から取り外したポイント情報を取り付ける。

【0047】一方、制御プログラム14によりCPU割当てを受け、配分中にゲストVMが待ち状態となった場合には、制御プログラム14はウェイトキュー24にレディキューの先頭ポイントからポイント情報を外して取り付けると同時に処理制御ブロック16の対応するゲストVMの配分比を残りの配分比となるように処理する。

【0048】このため、動作可能状態となって再度レディキュー18に取り付けられてCPU配分を受けた場合

には、待ち状態となったときに残っている配分比だけの再配分を行うことになる。図4は図3の仮想計算機システムにおけるCPU配分処理を示したフローチャートである。

【0049】図4において、まずステップS1で配分順序制御手段22がその時の処理制御ブロック16における動作カウンタ20の動作回数N1～N5と配分比を調べ、ゲストVM10-1～5の配分順序を決定し、レディキュー18に最初と最後のゲストVMのポインタ情報を取り付けており、このレディキュー18の先頭ポインタ情報を読み出してCPU配分を行うゲストVMを選択する。

【0050】続いてステップS2で選択したゲストVMにこのとき空き状態にある実CPU12-1または12-2を割当て、CPU配分を開始する。このCPU配分の実行中にあっては、ステップS3で現在CPU配分を受けているゲストVMの状態を監視しており、割当てられた実CPU12-1または12-2による処理時間が指定された配分比で規定される時間だけ使用されると、配分完了としてステップS4に進む。

【0051】ステップS4にあっては、動作カウンタ20の対応する動作回数に1を足す。続いてステップS5に進み、配分順序制御手段22がルール1及び2に従った優先度に従ってレディキュー18に対する割当て順番の再度取付けを行う。続いてステップS6に進み、後の説明で明らかにする待ち状態にあるゲストVMの中に動作可能なものがあるか否かチェックし、このとき待ち状態にあるゲストVMはないことから再びステップS1に戻り、レディキューから次のゲストVMを取り出して同様にCPU配分を行う。

【0052】一方、ステップS3でCPU配分を受けたゲストVMが配分比または配分処理の途中で待ち状態となった場合にはステップS8に進み、待ち状態における残り配分比、即ち残りの割当て時間を対応する処理制御ブロック(PCB)16の配分比の部分に格納する。続いてステップS9でウェイトキュー24に待ち状態となったゲストVMのポインタ情報を取り付ける。

【0053】ウェイトキュー24に対するポインタ情報の取り付けが済むとステップS5で配分順序制御手段22がルール1及びルール2に従って配分順序の再取付けを行い、ステップS6でウェイトキュー24に取付けたゲストVMの中で動作可能となったゲストVMがあるか否かチェックする。このとき、動作可能となったゲストVMがあった場合にはステップS7に進み、ウェイトキュー24のポインタ情報を取り外し、配分順序制御手段22のルール1及びルール2に従った優先度に従ってレディキュー18にウェイトキュー24から取り外したゲストVMのポインタ情報を取り付け、待ち状態から動作可能状態となったゲストVMに対するCPU配分の再開を可能とする。

【0054】図5は図3の実施例に示した本発明によるCPU配分の順番を示した説明図であり、図6に図5のCPU配分をCPU割当て時間の説明図として示す。図5のCPU配分処理はゲストVM10-1～5の名称をVM1～VM5で表し、配分比を

VM1	80%
VM2	80%
VM3	30%
VM4	5%
VM5	5%

に設定した場合である。初期状態において、動作カウンタ20のゲストVM1～VM5の動作回数N1～N5はそれぞれN1～N5=1となっている。従って、配分順序制御手段22はルール2に従った配分順序を設定する。即ち、ゲストVM1～VM5の配分比に従い、図7に示すように、配分比の大きい順番に並べる。

【0055】1回目のCPU配分の動作は実CPU12-2を優先順位の最も高いゲストVM1に割り付け、同時に実CPU12-2を次に優先順位の高いゲストVM2に割り付け、ゲストVM1とVM2の配分処理を同時に開始する。ゲストVM1とゲストVM2の配分比は同じ80%であることから、図6から明らかなように実CPU12-1、12-2によるゲストVM1、VM2の配分処理は同時刻に終了する。

【0056】続いて実CPU12-1を3番目に優先度の高いゲストVM3に割り付け、同時に実CPU12-2を4番目に優先度の高いゲストVM4に割り付け、同時に配分処理を開始する。実CPU12-2を割り付けたゲストVM4の配分比は5%と小さいため、実CPU12-1を割り付けた配分比30%のゲストVM3の配分処理が終了する前にゲストVM4の配分処理が終了する。ゲストVM4の配分処理が終了すると実CPU12-2をゲストVM5に割り付け、ゲストVM5の配分処理を開始する。ゲストVM5の配分処理が終了しても、このときゲストVM3はまだ配分処理中にある。

【0057】このため、ゲストVM5の配分処理が終了すると実CPU12-2をゲストVM1に割り付け、2回目の配分処理を開始し、ゲストVM1の配分処理中にゲストVM3の1回目の配分処理が終了する。1回目のゲストVM1～VM5の配分処理が終了するごとに動作カウンタ20の動作回数N1～N5は1ずつカウントアップされ、次のCPU配分を開始する際に、配分順序制御手段22がルール1及び2に従った優先順位に基づく配分順の設定を行う。

【0058】このため、第1回目の配分処理において、ゲストVM4とVM5の配分処理がゲストVM3の配分処理より前に終了していても、最後にゲストVM3の配分が終了してゲストVM1～3の1回目の配分処理が終了した段階で動作カウンタ20の動作回数N1～N5は全て同じ値となっており、動作回数N1～N5が同じ場



合にはルール2による配分比の大きい順に並べることから、2回目の配分処理を開始した段階でも1回目と同様、ゲストVM1～VM5の配分順位は図7に示した最初と同じ状態を維持する。

【0059】従って、配分処理を繰り返しても処理制御ブロック16で指定した配分比を維持することができ、実CPU12-1, 12-2を効率よく利用することができる。図8はゲストVM1～5の配分処理の途中でゲストVM2に待ち状態が生じたときの説明図である。

【0060】図8において、1回目の配分処理でゲストVM1～VM5の順番で配分処理を行っているとき、ゲストVM2が配分比10%に相当する時間、配分処理を行ったときに待ち状態を生じたとする。このようにゲストVM2が待ち状態になるとそのポインタ情報がウェイトキュー24に取り付けられ、同時に制御処理ブロック16のゲストVM2の配分比を残り70%に設定する。

【0061】ゲストVM2の待ち状態への以降処理が済むと次のゲストVM3の配分処理に移行する。1回目の配分処理がゲストVM5間で済んだ状態で待ち状態にあったゲストVM2が動作可能状態になったとすると、そのポインタ情報をウェイトキュー24から取り外してレディキュー18に取り付ける。このとき、レディキューにはすでに1回目の割り付け処理が済んで2回目の割り付け処理待ち状態にあるゲストVM1及びVM3～VM5が並んでいるが、待ち状態におかれたゲストVMの動作回数N2はN2=1と2回目の配分処理を待っているゲストVM1およびVM3の動作回数N1, N3～N5=2より小さいことから、動作可能状態となったゲストVM2の残りの配分比70%の配分処理が優先して行われる。

【0062】このため、待ち状態にあったゲストVM2の1回目の配分処理が完了すると2回目の割り付け処理の順番は全ての動作回数N1～N2=2となっていることから、配分比の大きい順番に並べられ、図7に示した順番を維持した配分処理が繰り返される。このため、特定のゲストVMが待ち状態となって一時的に配分順序が乱れても、次の配分処理の際には正しい順番に戻った配分が行われることになる。

【0063】尚、上記の実施例は2つの実CPUを5つのゲストVMに割り付けてCPU資源の配分処理を行う場合を例にとるものであったが、本発明はこれに限定されず、任意の数の実CPUを同じく任意の数のゲストVMに割り付ける配分処理にそのまま適用することができる。

【0064】

【発明の効果】以上説明してきたように本発明によれば、指定された配分量を消費するごとに1つカウントアップされる動作カウンタの動作回数を用いた配分順序の優先制御を行うことによって複数の実CPUを割り付けた複数のゲストVMの配分処理において指定された配分

量の通りに正確に配分処理を繰り返すことができ、実CPU資源を有効利用した配分処理による仮想計算機システムの動作ができる。

【0065】また、CPU配分を受けたゲストVMが待ち状態となっても次のCPU配分では元の配分順序に戻ることでゲストVMの待ち状態の発生により配分を乱されることなく実CPU資源を有効に利用することができる。

【図面の簡単な説明】

【図1】本発明の原理説明図

【図2】本発明のハードウェア構成を示した実施例構成図

【図3】本発明によるCPU資源の配分処理の一実施例を示した説明図

【図4】本発明によるCPU資源の配分処理を示したフローチャート

【図5】本発明による配分比が不均等な場合の配分処理の順番を示した説明図

【図6】図5の配分処理をCPUの割当時間で示した説明図

【図7】本発明で全てのゲストVMが動作可能な時のCPU配分の順番を示した説明図

【図8】配分中に待ち状態が起きた場合の配分処理を示した説明図

【図9】従来の単一の実CPUを用いた仮想計算機システムの説明図

【図10】図9の単一実CPUにおけるCPU配分処理を示したフローチャート

【図11】従来の配分比に従った配分順を示した説明図

【図12】従来の複数の実CPUを用いた仮想計算機システムの説明図

【図13】複数の実CPUを用いた従来の均等配分の処理説明図

【図14】複数の実CPUを用いた従来の不均等配分の処理説明図

【図15】図14の不均等配分をCPU割当時間で示した説明図

【図16】図14の不均等配分の繰り返して飽和状態に至った時の配分比の指定量と実際の割当量を示した説明図

【符号の説明】

10-1～10-n：仮想計算機（ゲストVM）

12-1, 12-2：CPU（実CPU）

14：制御プログラム（CP）

16：CPU割当情報域（処理制御ブロック；PCB）

18：レディキュー

20：動作カウンタ

22：配分順序制御手段

24：ウェイトキュー

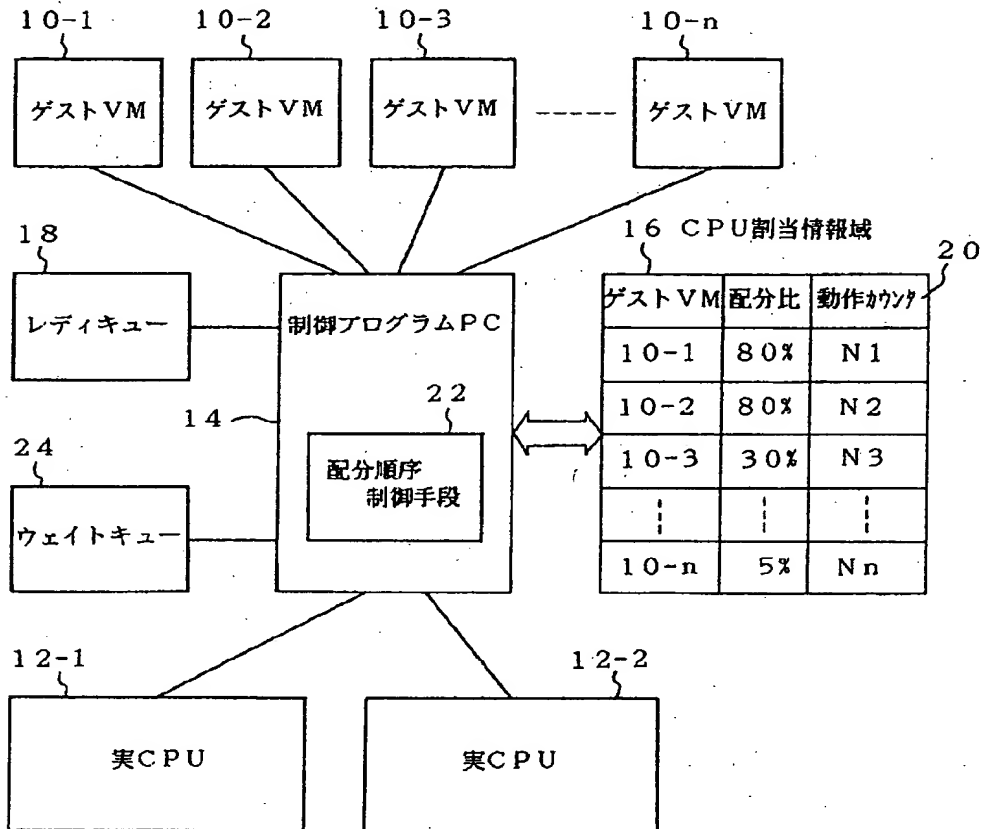
26：主記憶制御装置（MCU）

28 : 主記憶装置( MS U )  
 30 : オペレーティングシステム( OS )  
 32 : チャンネルプロセッサ( CHP )

34 : デバイスバス  
 36 - 1 , 36 - 2 : デバイスインタフェース  
 38 : 磁気ディスク装置

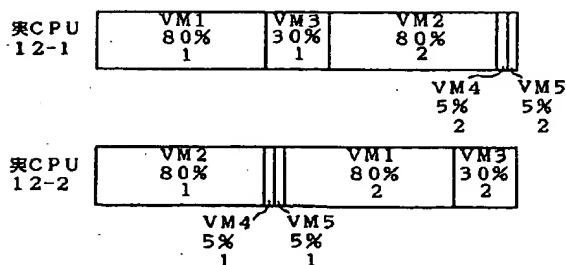
【 図1 】

## 本発明の原理説明図



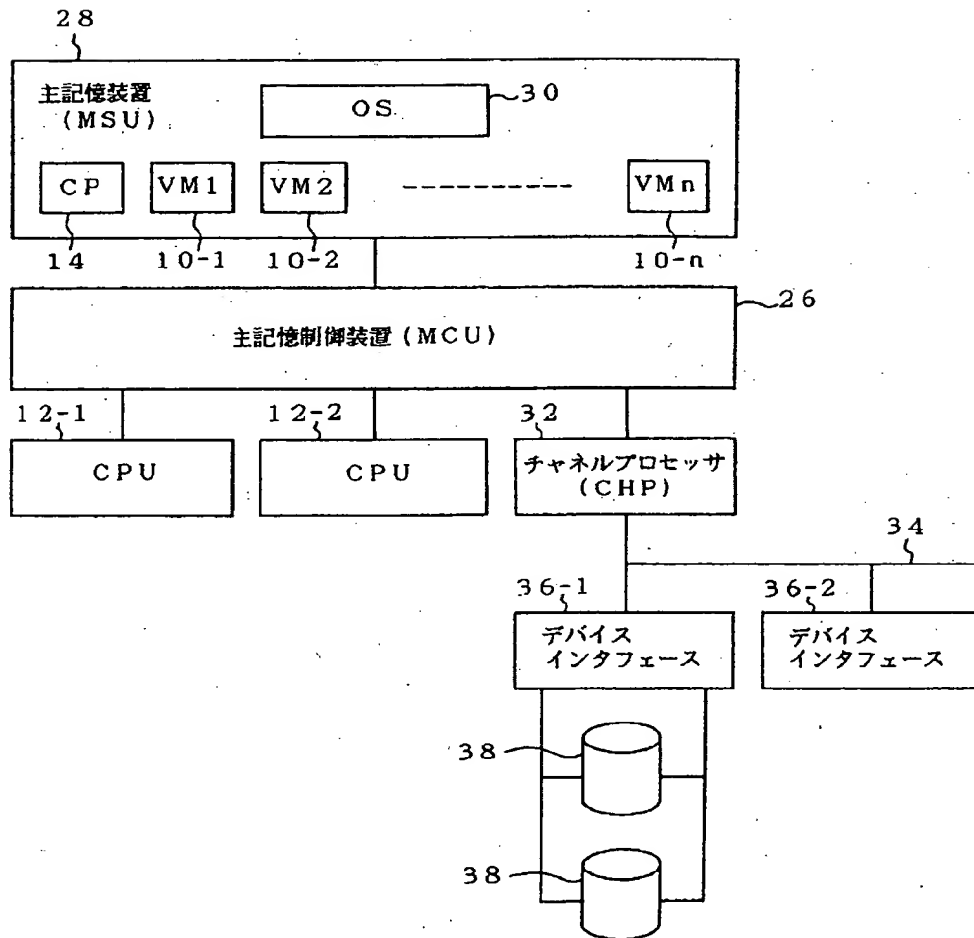
【 図6 】

図5の配分処理をCPUの割当時間で示した説明図



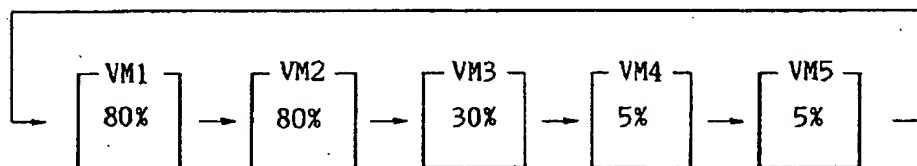
【 図2 】

本発明のハードウェア構成を示した実施例構成図



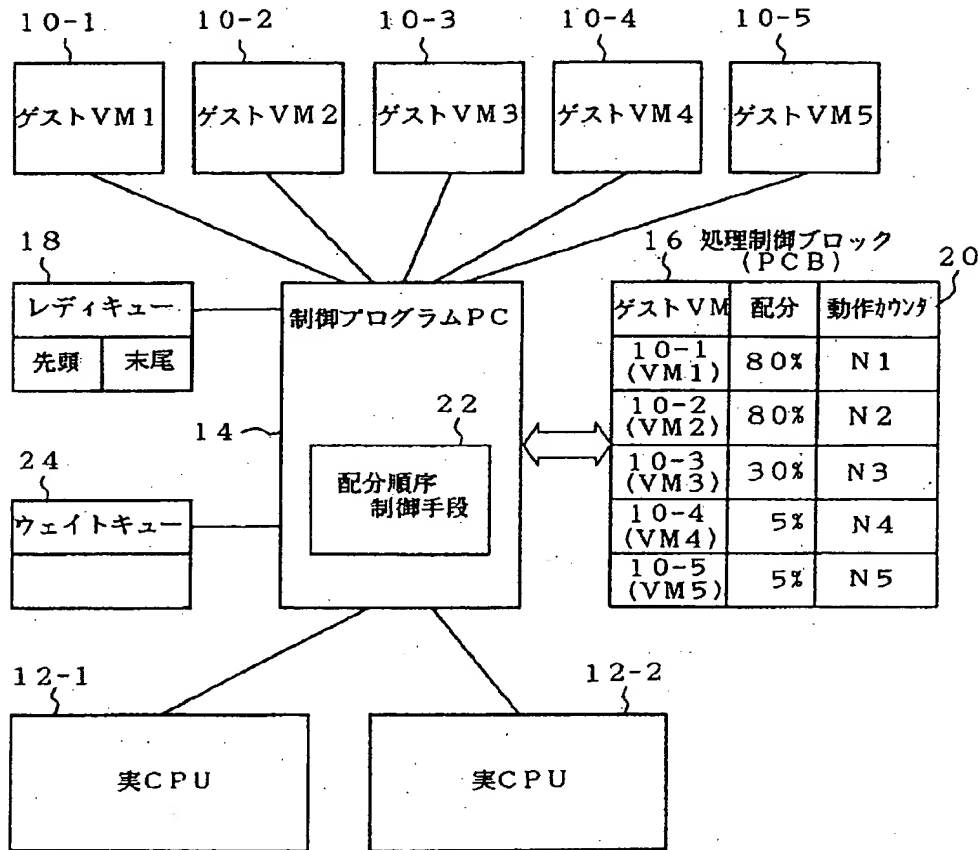
【 図7 】

本発明で全てのゲストVMが動作可能な時のCPU配分の順番を示した説明図



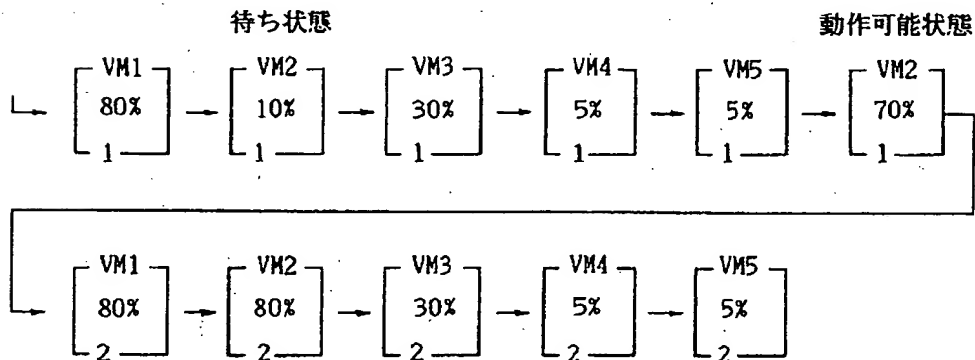
【 図3 】

本発明によるCPU資源の配分処理の一実施例を示した説明図



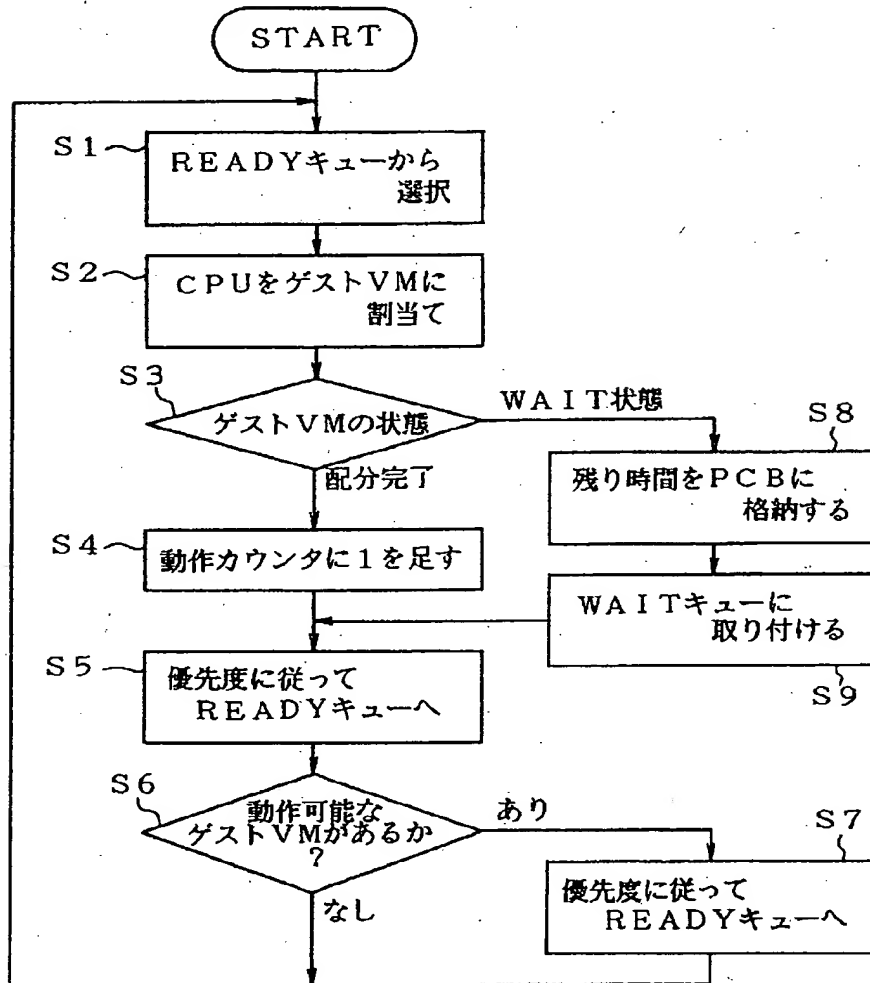
【 図8 】

配分中に待ち状態が起きた場合の配分処理を示した説明図



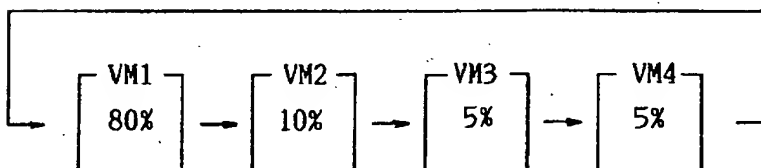
【 図4 】

本発明によるCPU資源の配分処理を示したフローチャート

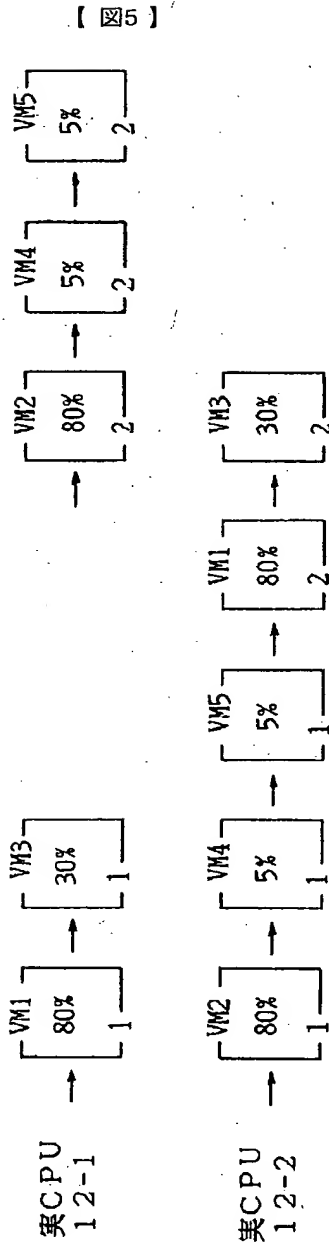


【 図11 】

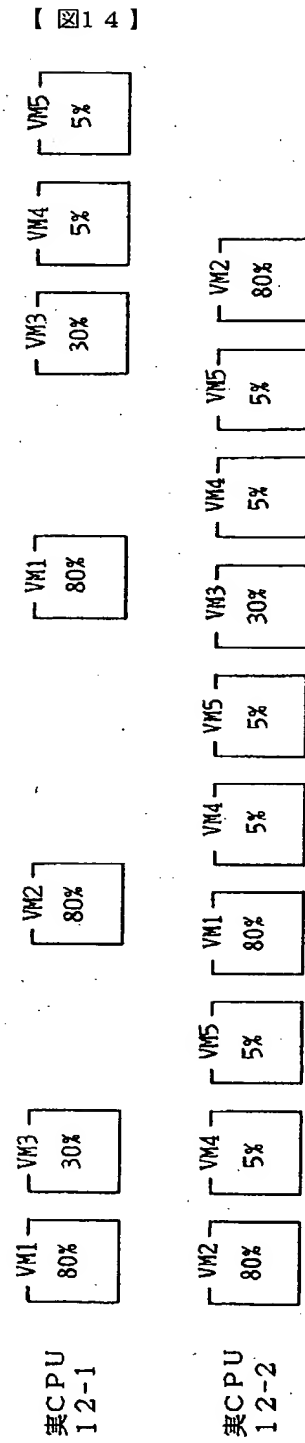
従来の配分比に従った配分順を示した説明図



本発明による配分比が不均等な場合の配分処理の順番を示した説明図

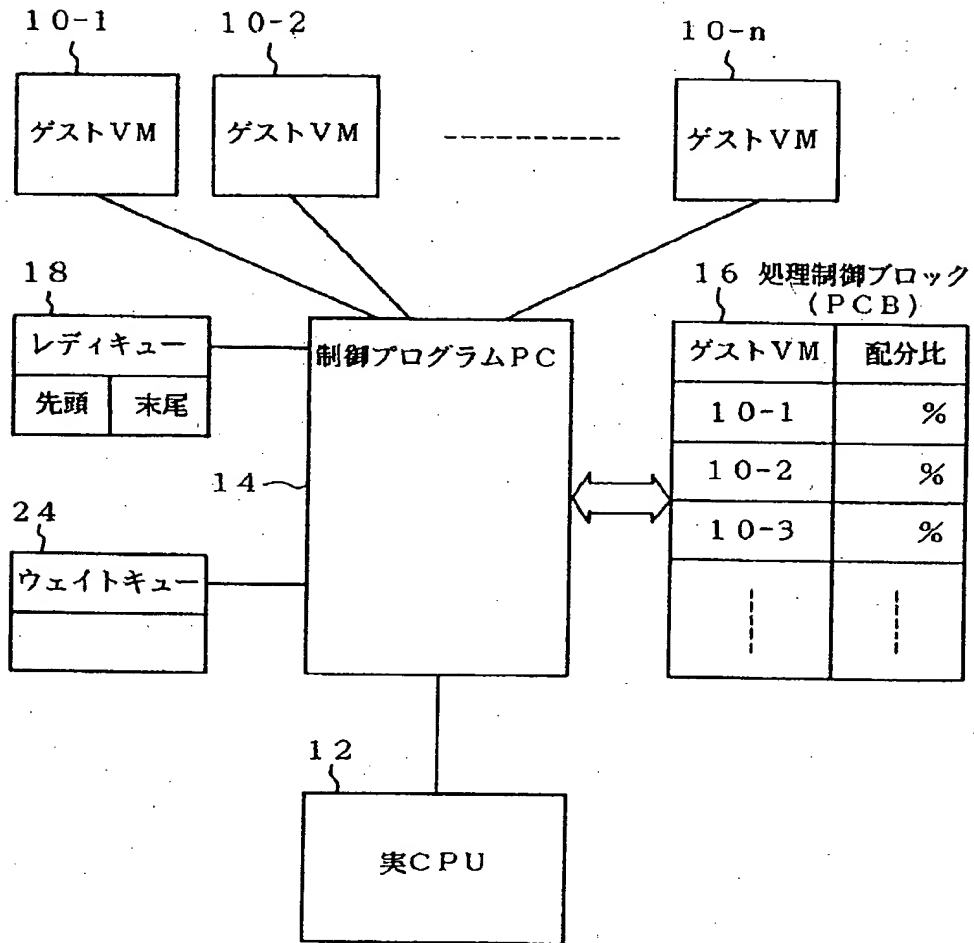


複数の実CPUを用いた従来の不均等配分の処理説明図



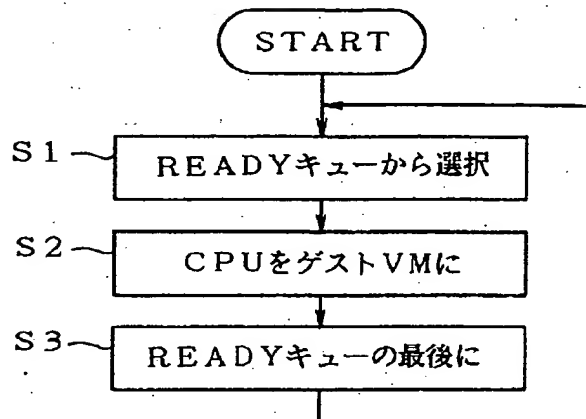
【 図9 】

従来の単一の実CPUを用いた仮想計算機システムの説明図



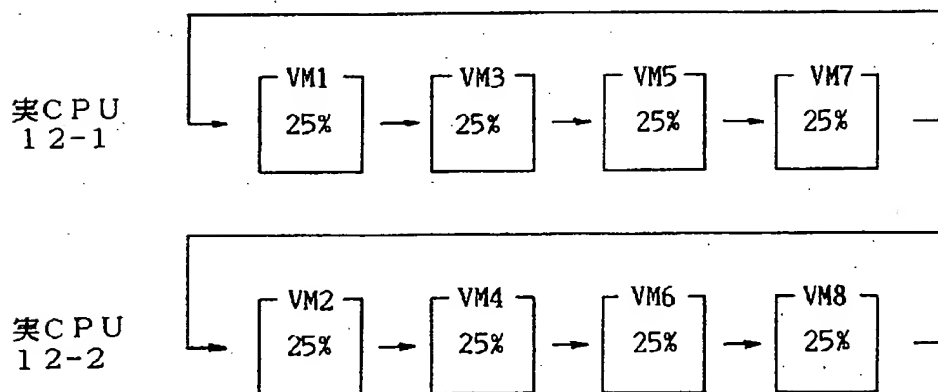
【 図1 0 】

図9の単一実CPUにおけるCPU配分処理を示したフローチャート



【 図1 3 】

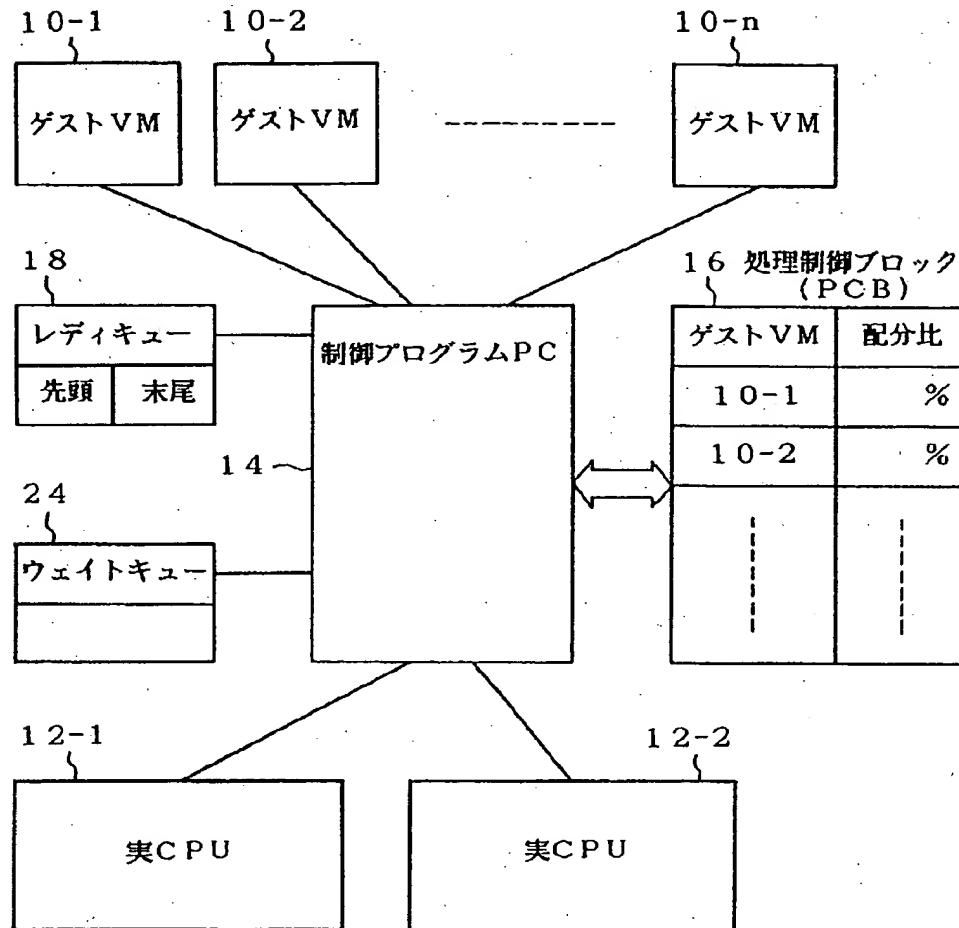
複数の実CPUを用いた従来の均等配分の処理説明図





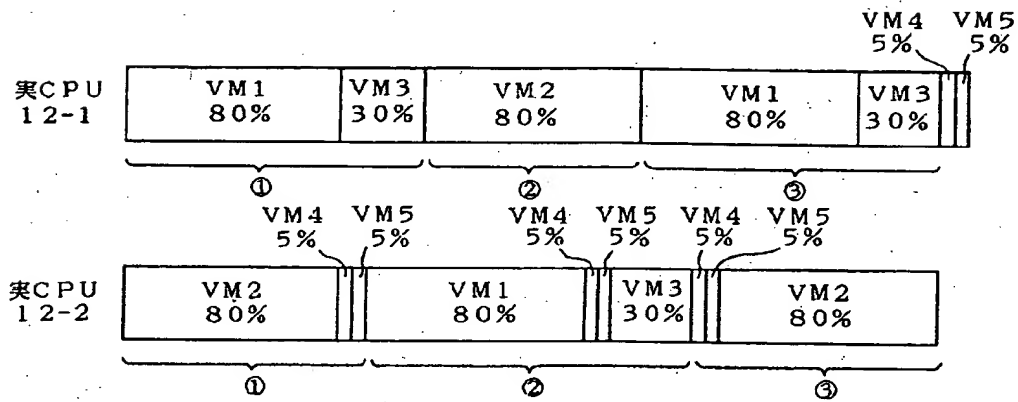
【 図1 2 】

従来の複数の実CPUを用いた仮想計算機システムの説明図



【 図1 5 】

図1 4の不均衡配分をCPU割当時間で示した説明図



【 図1 6 】

図1 4の不均衡配分の繰り返して飽和状態に至った時の配分比の指定量と実際の割当量を示した説明図

VM名	割当量	指定量
VM1	78	80
VM2	78	80
VM3	28	30
VM4	7	5
VM5	7	5